P516704.PDF [Page: 3 of 31]

Copy No. 28

DEFENCE R&D DÉFENSE

DREA IDL LIBRARIAN USER'S GUIDE

Final Report

C. Calnan
xwave
5475 Spring Garden Road
Suite 601
Halifax, Nova Scotia B3J 3T2

Defence R&D Canada

DEFENCE RESEARCH ESTABLISHMENT ATLANTIC

Contractor Report

DREA CR 2000-167

September 2001

National Defence Défense nationale

Canada

P516704.PDF [Page: 4 of 31]

104-07010/

Copy No: _____

DREA IDL Librarian User's Guide

Final Report

Colin Calnan
xwave
5475 Spring Garden Road
Suite 601
Halifax, Nova Scotia B3J 3T2

The scientific or technical validity of this Contractor Report is entirely the responsibility of the contractor and the contents do not necessarily have the approval or the endorsement of Defence R&D Canada

Scientific Authority. James A. Theriault

Defence Research Establishment Atlantic

Contract Report
DREA CR 2000-167
September 2001

James A. Theriault
Scientific Authority

Approved by

David M.F. Chapman
Head/Naval Sonar Section

Approved for release by

Kirk Foster
Head/Document Review Panel

© Her Majesty the Queen as represented by the Minister of National Defence, 2001
© Sa majesté la reine, représentée par le ministre de la Défense nationale, 2001

P516704.PDF [Page: 6 of 31]

Abstract

Over the years a not inconsiderable amount of IDL code has been written to perform data analyses and produce graphic output. However, the vast majority of the IDL code has been written almost in isolation, with the various programmers generally working with little contact with one another. The result is that the same programs or functions are continually being rewritten as programmers are unaware that programs that they working on have already been written.

To circumvent this duplication of effort a contract was awarded to produce a library of IDL code. After consultation with various IDL programmers and considering the options, it was decided to make the library available via the DREA Intranet, and searchable in a number of ways. This was done.

Creating a library, however, is only the start of the process. The ongoing work involved with any library is maintenance, wherein the contents are kept up to date and new entries are made accessible to users as soon as possible. This manual describes the tasks required of an IDL Librarian in order to keep the library current.

Résumé

Avec les années, une quantité non négligeable de code IDL a été préparée pour des analyses de données et la production de sorties graphiques. La plus grande part du code IDL a toutefois été préparée isolément, les divers programmeurs travaillant en général sans beaucoup de contacts entre eux. Ne sachant pas que les programmes qui'ls développent existent déjà, les programmeurs travaillent constamment aux mêmes programmes ou fonctions.

Afin de remédier à ce dédoublement d'efforts, un contrat a été accordé dans le but de produire une bibliothèque de code IDL. Après la consultation de divers programmeurs en IDL et l'examen des options possibles, il a été décidé de rendre cette bibliothèque accessible sur l'intranet du CRDA et d'en permettre la consultation d'un certain nombre de façons. C'est ce qui a été fait.

La création d'une bibliothèque ne représente toutefois que le début du processus. Toute bibliothèque exige des travaux de maintenance, garantissant la tenue à jour du contenu et l'accessibilité la plus rapide possible aux nouvelles entrées. Ce manuel décrit les tâches que doit effectuer un bibliothécaire IDL pour tenir la bibliothèque à jour.

P616794.PDF [Page: 7 of 31]

This page intentionally left blank.

Page laissée en blanc intentionnellement.

P516704.PDF [Page: 8 of 31]

Executive summary

Scientists at the Defence Research Establishment Atlantic (DREA) have undertaken activities involving the analysis of acoustic data. In many cases, these researchers have developed their own tools to analyze and visualize this data. Through this approach, DREA has made significant investment in software tools, but little effort has gone into managing those tools.

Many of the tools have been developed using the commercial IDL (Interactive Display Language) software package. Many scientists have offered to contribute the software developed for various projects to a consistently managed library to be available to all DREA staff.

Such a library of IDL programs was created as a deliverable for this contract. After consultation with various IDL programmers and considering the options, it was decided to make the library available via the DREA Intranet, and searchable in a number of ways. The initial library was created, but is not fully populated.

Creation of the library is only the start of the process. The ongoing work involved with any library is maintenance, wherein the contents are kept up to date and new entries are made accessible to users as soon as possible.

This DREA contract report describes the IDL library infrastructure, and the tools required by the IDL Librarian in order to keep the library current.

Calnan, C.; 2001; DREA IDL Librarian User's Guide, Final Report;
DREA CR 2000-167; Defence Research Establishment Atlantic.

P516704.PDF [Page: 9 of 31]

Sommaire

Des scientifiques du Centre de recherches pour la défense Atlantique (CRDA) ont entrepris des activités nécessitant l'analyse de données acoustiques. Dans bien des cas, ces chercheurs ont mis au point leurs propres outils d'analyse et de visualisation des données. En procédant ainsi, le CRDA a largement investi dans des outils logiciels, mais peu d'efforts ont été consacrés à la gestion de ces outils.

Beaucoup des outils ont été réalisés à l'aide du logiciel IDL (langage d'affichage interactif), disponible sur le marché. De nombreux scientifiques ont offert d'intégrer des logiciels produits pour divers projets à une bibliothèque gérée de façon continue et accessible à tout le personnel du CRDA.

Cette bibliothèque de programmes IDL a été créée en tant que produit livrable de ce contrat. Après la consultation de divers programmeurs en IDL et l'examen des options possibles, il a été décidé de rendre la bibliothèque accessible sur l'intranet du CRDA et d'en permettre la consultation d'un certain nombre de façons. La bibliothèque initiale a été créée, mais elle n'est pas totalement remplie.

La création d'une bibliothèque ne représente toutefois que le début du processus. Toute bibliothèque exige des travaux de maintenance, garantissant la tenue à jour du contenu et l'accessibilité la plus rapide possible aux nouvelles entrées.

Ce rapport de contrat du CRDA décrit l'infrastructure de la bibliothèque IDL ainsi que les outils dont le bibliothécaire IDL a besoin pour tenir la bibliothèque à jour.

Cainan, C.; 2001; DREA IDL Librarian User's Guide, Final Report;
DREA CR 2000-167; Defence Research Establishment Atlantic.

P516704.PDF [Page: 10 of 31]

Table of Contents

1.0 INTRODUCTION 1

2.0 IDL LIBRARY ROOT DIRECTORY 2

3.0 COMMENT BLOCKS AND INFORMATION TAGS 4

4.0 IDL LIBRARY SUBMISSION FILE TYPES 5
4.1 IDL Code 5
4.2 Referral Files 5

5.0 PUTTING FILES INTO THE IDL LIBRARY SYSTEM 6
5.1 File Preparation and Placement 6
5.2 CVS Version Management 6
5.3 Building the Library Index 8
5.4 Creating Downloadable Files 9

6.0 RETRIEVING OLD CODE 10

7.0 SUGGESTIONS FOR FURTHER WORK 11

APPENDIX A. INFORMATION TAGS 12

APPENDIX B. **MakeFI.pl** AND **file.info** CONTENTS 14

APPENDIX C. USER ROUTINES 16

P516704.PDF [Page: 11 of 31]

1.0 INTRODUCTION

A library has been created that contains IDL programs developed and used at DREA. The library is accessible to users through DREA's Intranet, and it provides users with the ability to search for specific IDL files, routines, or even text within the routines. Search results allow users to read descriptions and information on the files (if any) and download or otherwise access the files that satisfy the search parameters.

As might be expected, the maintenance of the library does not come without some overhead. As old files are modified or new ones created, they must be put into the library, a task that falls to an IDL Librarian. This report describes how the IDL Librarian is to perform the tasks required to keep the library up to date. As well, several appendices contain information on some of the more "hidden" parts of IDL Library maintenance. These appendices are included for completeness and in case the functionality of the IDL Library is to be expanded.

Because the IDL Librarian's primary task is to keep the library current, files submitted to the librarian should be entered into the library system as soon as possible. Part of this means that the supplementary files used transparently by the users are kept current as well. This document provides instructions on how these tasks may be performed.

The term "Library code" is occasionally used in this document. It refers to the code (HTML and Perl scripts mostly) used by the IDL Library system to keep things in order, produce indexing and downloadable files, communicate with users, etc. It does not refer to the IDL code that is stored in the IDL Library.

Besides common English typographic conventions, the following conventions are used in this document:

- **bold text** is used for filenames (e.g. **Program.f** or **Fozzie:/local/files/Program.f**)
- ***bold italics text*** is used for directories (e.g. ***/usr/tmp*** or ***Fozzie:/local/files***)
- *italics text* is used for computer names (e.g *Fozzie*)
- `Courier text` is used for text to be typed on the keyboard (e.g. enter "`idl`")

P516704.PDF [Page: 12 of 31]

2.0 IDL LIBRARY ROOT DIRECTORY

The librarian performs all tasks on the computer *WWW*, logging into account *idllib* and then entering "`cd $IDLLIB`" to get to the library root directory. That directory is currently */users/htdocs/DREA/common/idl/IDL-Library* and the environment variable IDLLIB has been set to that directory in the account's **.login** file. The library root directory will be referred to as ***$IDLLIB*** in this report.

The ***$IDLLIB*** files of import are:

file.info	- the index of all the IDL code in the library; details on this file can be found in Appendix B
tg.sh	- a shell script created at the same time as **file.info**
dir.nolook	- a file listing directories that are not to be searched for IDL code (the library's root directory is included in this list by programming default). The directories listed in this file contain the CGI scripts and HTML documents used by the system, test code, contributed files that have not yet been integrated into the system, and other non user-related material. These directories are included in **dir.nolook** if they contain **.pro** files that are not to be part of the IDL Library (either at all or in their current location) or if they don't contain any **.pro** files and searching through them would just be a waste of time.
MakeFI.pl	- a Perl script that reads **dir.nolook** and creates **file.info** and **tg.sh**; see Appendix B for more on this script
.cvsignore	- a list of files in the current directory that are not to be included in the CVS system, when it is put in place; every directory from ***$IDLLIB*** on down should have a file by this name, and they should all contain (at the least) "`*.tar.gz`"
other files	- anything else in here was probably involved in the system's development as a prototype, test routine, etc.; in any event they are of no interest at this point
NO IDL!!!	- there must be no IDL code in ***$IDLLIB***; this is mandated both to help keep the directory clean and because the Library code does not look there for code and so will overlook it

Some of the directories located in ***$IDLLIB*** are:

CGI	- contains the CGI Perl scripts used by the system
html	- contains the HTML documents used by the system
OldStuff	- contains development code and materials, as well as outmoded routines that may have something useful for the future
contrib	- a temporary home for new routines; this directory is referred to in Section 5

All other directories off ***$IDLLIB*** contain the IDL code in whatever sort of tree structure is deemed useful by the librarian. For the moment, most of these directories are named either after the IDL programmer who has submitted the code that the directories contain, or after the main program in the directory. If the owner of some IDL code already has a

P516704.PDF [Page: 13 of 31]

particular directory structure set up for storing his code, then that structure will probably be used in the library.

3.0 COMMENT BLOCKS AND INFORMATION TAGS

The term "comment block" is used freely in this document, and a description is in order. A comment block is a group of lines in IDL code (or actually in any ASCII file) that follows the line ";+" and continues until the line ";-" is found. Comment blocks are based on a similar fancy in IDL where all the lines between ";+" and ";-" are turned into HTML documentation via the command "`MK_HTML_HELP`". The lines in a comment block normally start with ";", which indicates the start of a comment in IDL, hence the term comment block, but strictly speaking the lines in a comment block don't have to be comments. If a comment block contains nothing but lines starting with ";", then this document may refer to them as "pure comment blocks."

IDL Library code has been written that extracts comment blocks from files and presents them to users in HTML documents, with the intention that comment blocks be used to describe and explain programs. As this is the only way that the system can extract and present file and routine information, it is strongly recommended that comment blocks be present in all files, as recommended in the following section.

Appendix A describes the information tag scheme used by the IDL Library system. But in summary, information tags are lines in a comment block that start with ";>>*[information tag]*:", which is followed by the tag's information on the same line. The current information tags are "IDLLIB Date", "Name", "Access", "Files Needed", "Owner", "Last Update", "Version", and "Keywords". The first tag named is needed for CVS usage, and then Library code only exists to use the next four tags, although **MakeFI.pl** reads them all and puts their contents into **file.info**. Section 4 and its subsections refer to a few of these information tags, mainly those that are mandatory, but it would be most beneficial if all the tags were included wherever possible. Of these tags, only the "Owner" is currently displayed explicitly in users' search results.

Many programmers like to surround their comments with boxes of various symbols (asterisks, hash symbols, exclamation marks, etc.), so the Library code was written to consider this when decyphering information tags. It simply ignores these characters if they appear by themselves and doesn't take them to be part of the information tags' data. However the Library code does take letters and numbers to be valid data, so these characters can't be used to block off the comments.

P516704.PDF [Page: 15 of 31]

4.0 IDL LIBRARY SUBMISSION FILE TYPES

The librarian may receive two different types of files for inclusion in the library: IDL code and referral files. The files should be examined and pre-processed before being put into the library, with the amount and type of pre-processing depending on the files' types. This section describes the file types and explains the sort of pre-processing that is required for each type.

4.1 IDL Code

The first submission type is straightforward IDL code, which is always an ASCII file with the extension **.pro** in any mix of cases (e.g. **fubar.pro** or **APROG.PRO**). Strictly speaking, IDL code does not have to be changed at all before being put into the system. However, if it contains a group of comment lines near the top (or anywhere else, for that matter) the comments may be turned into a comment block by putting the line ";+" before the comments start and the line ";-" after the comments end. IDL Library code has been written to extract comment blocks for a user's perusal.

If the submitted program requires other files in order to run, then the file information tag ";>>Files Needed:" should be included in the comment block, followed by a list of the extra files. This is the only way that the library system can know about other required files, and the tag should not be omitted if extra files are required.

It may happen that IDL code requires non-IDL files to run (for example, the IDL SAPLOT program needs the file **idl-saplot.help**, an informational help file). When this occurs the non-IDL file can be submitted as is as long as two conditions are met: it must be an ASCII file, and it must be listed after the ";>>Files Needed:" information tag in the appropriate IDL file(s).

4.2 Referral Files

A user may have a program and not wish to allow it to be freely available. If the user were still amenable to having others use his program, albeit under his control, he could submit a description of his program. This "referral file" must:

- have the same name as the program it is a referral for
- have the filename extension **.pro**
- be a pure comment block
- not contain the ";>>Files Needed:" tag unless that tag is present without any parameters
- have the ";>>Access:" tag followed by the e-mail address of the program's owner

The IDL Library code can therefore locate referred programs in searches via their referral files, and the ";>>Access:" tag allows the Library programs to refer an interested user to the programs' owners by their e-mail addresses.

P516704 PDF [Page: 16 of 31]

5.0 PUTTING FILES INTO THE IDL LIBRARY SYSTEM

This section details what the IDL Librarian must do to put the submitted files into the IDL Library system.

5.1 File Preparation and Placement

The Librarian must deal with submissions according to their types, as indicated in the previous section. This can be done either immediately on the librarian's PC, wherever the e-mail submissions are received, or later in the IDL Library. Once files, in whatever state of readiness they possess, can be sent to the Library, they should be placed in the directory ***$IDLLIB/contrib*** by any means that the Librarian likes. (The Librarian can feel free to create subdirectories off ***$IDLLIB/contrib*** if it seems appropriate.) If the files don't yet follow the requirements of the previous section, they can be edited into compliance in this directory.

Once the files are compliant, they must be moved to an appropriate directory off ***$IDLLIB***. If a submitted file is a newer version of a file that already exists in that directory, then the older version may be deleted. The last preparation step is to edit the ";>>IDLLIB Date:" information tag to contain the current date (see Appendix A for more on this and other information tags, including the expected date format).

All files to be placed in the IDL Library must be plain text files with the carriage return/line feed end-of-line demarcation required by the host computer. Files are not allowed to contain non-text characters as this may confuse CVS and make it impossible to retrieve old files.

5.2 CVS Version Management

The IDL code in the library is placed under CVS control in case a user wants to retrieve an older version of a specific file. There may be a number of reasons that a user would want an older version of a program: a newer version may have bugs, a newer version may have had a desired feature removed, the older version may read a different data format, etc.

The date in the information tag ";>>IDLLIB Date:" must be the date that the file is placed into CVS. If it isn't, then it will be very difficult to retrieve old files. For the purposes of this *User's Guide* it is assumed that the tag's date is correct, and no instructions are given on what to do if that information tag's date is incorrect. That information tag must also be put into all non-IDL code files listed under the ";>>Files Needed:" information tags in any IDL code.

Before anything can be done in CVS, the IDL Librarian must log into CVS. This is done by entering "`cvs login`" at a UNIX prompt, which will cause the program to print the

CVS prompt "CVS password:". Enter the password used to log into the account *idllib*.

Then, putting code into CVS is done under one of four cases, each of which has its own procedure that must be followed.

Case 1: New directory ***newdir*** off ***$IDLLIB***

```
cd $IDLLIB/newdir
echo "*.tar.gz" > .cvsignore
cvs import -m "Initial" $IDLLIB/newdir vtag rtag
cd $IDLLIB
mv newdir newdir-orig
cd ..
cvs checkout $IDLLIB/newdir
```

The IDL Librarian should then do spot checks to verify that the contents of directories ***newdir*** and ***newdir-orig*** are identical as far as the IDL files are concerned. When this has been done the entire directory ***newdir-orig*** and its contents must be deleted.

Case 2: New directory ***newdir*** off existing directory ***$IDLLIB/old1/old2***

```
cd $IDLLIB/old1/old2/newdir
echo "*.tar.gz" > .cvsignore
cvs import -m "Initial" $IDLLIB/old1/old2/newdir vtag rtag
cd ..
mv newdir newdir-orig
cd $IDLLIB
cd ..
cvs checkout $IDLLIB/old1/old2/newdir
```

The IDL Librarian should then do spot checks to verify that the contents of directories ***$IDLLIB/old1/old2/newdir*** and ***$IDLLIB/old1/old2/newdir-orig*** are identical as far as the IDL files are concerned. When this has been done the entire directory ***$IDLLIB/old1/old2/newdir-orig*** and its contents must be deleted.

Case 3: The new file **newfile.pro** has been put into the CVS directory ***$IDLLIB/old***

```
cd $IDLLIB/old
cvs add newfile.pro
cvs commit -m "Initial" newfile.pro
mv newfile.pro newfile.pro-orig
cd $IDLLIB
cd ..
cvs checkout $IDLLIB/old/newfile.pro
```

P516704 PDF [Page: 18 of 31]

The IDL Librarian should then compare the files ***$IDLLIB/old/newfile.pro*** and ***$IDLLIB/old/newfile.pro-orig*** to ensure that they are identical. When this has been done ***$IDLLIB/old/newfile.pro-orig*** must be deleted.

Case 4: Some revised files have been put into the CVS directory ***$IDLLIB/old***

Make sure that the revised version has a different information tag ";>>Last Update:" date than the older version in all revised files. If version numbers are used in the files, make sure that they are updated as well. The procedure indicated below will put all the revised files, and only the revised files, into the CVS system.

```
cd $IDLLIB/old
cvs commit -m "Revised"
```

The changed files will be written to the screen by CVS. Unless it was already known exactly which files have been revised, their names should be noted as they scroll by. Then for each of the revised file enter:

```
mv revised.pro revised.pro-orig
```

Once this is done, continue with:

```
cd $IDLLIB
cd ..
cvs checkout idllib/old
```

The original files (i.e. those whose names now end with "**orig**") will be flagged with "?" in column 1 as written to the screen; this is normal and indicates that the files are not in CVS. The revised files will be flagged with a "U" in column 1, to show that they have been updated.

The IDL Librarian should check that the "**orig**" files are identical with their non-"**orig**" versions. After this has been done, the "**orig**" versions are to be deleted.

5.3 Building the Library Index

After the files have been put into CVS the Librarian must go to ***$IDLLIB***. There the file **file.info** should be renamed to a backup name since a new version will be produced.

The Librarian must then run the routine **MakeFI.pl**, which can be done by entering the routine's name at the UNIX prompt. At the end of its run, the program will write some status information to the screen. If there were problems with any of the **.pro** files the Librarian will be told to check out the file **file.errors**. That file lists the files that **MakeFI.pl** has problems with and gives details on the problems. All files listed in **file.errors** are omitted from **file.info**, so it is important that there be no errors in any of the **.pro** files. Once **MakeFI.pl** can run without error, the file **file.info** contains the complete information on all the ***$IDLLIB*** **.pro** files (or at least those in directories not named in the file **dir.nolook**).

The Librarian may want to run a difference on **file.info** and its previous version. This may be done to see if the differences in the file accurately reflect the changes known to have been made in the contents of the Library. This non-essential procedure is a just-in-

case check and is not really required. Once the Librarian is content with the **file.info** contents, the old version may be deleted.

As originally set up, the directory ***$IDLLIB/contrib*** is included in **dir.nolook**. This means that none of the files in that directory will be included in **file.info** and so be available to the user. It is obvious that if that directory is removed from **dir.nolook** then the ***contrib*** files will be included in **file.info**, but this action will no doubt lead to complications. In order to further discourage this action, none of the complications will be listed in this guide, and neither will the ways to clear them up.

5.4 Creating Downloadable Files

MakeFI.pl also tells the Librarian to run the Korn shell script **tg.sh**, which can be done by entering the command "`ksh tg.sh`". This script performs a **tar/gzip** combination on all the **.pro** files in the Library, producing the files that users download. **tg.sh** should only be run after a clean run of **MakeFI.pl** has been performed.

Once all the files have been cleared from ***$IDLLIB/contrib*** and **MakeFI.pl** and **tg.sh** have been run, the Librarian's work is done until the next batch of files are to be entered into the system.

P516704.PDF [Page: 20 of 31]

6.0 RETRIEVING OLD CODE

It may happen that a user wishes to retrieve an old version of some IDL code. The only way that this can be simply done is if the user knows when the code was put in the IDL Library. The date of that entry is indicated with the file's ";>>IDLLIB Date:" information tag, as is mentioned in Section 5.2 and Appendix A. It is hoped that the user would be aware of this date, and this section was written under the assumption that the date is known. If the date is not known, then the task becomes more difficult, and someone knowledgeable in the ways of CVS will have to be searched out in order to obtain the desired code.

Before any CVS work can be done, however, the IDL Librarian must log into CVS. Section 5.2 has instructions on how this may be done.

For the sake of this Section's example, we are assuming that we want to retrieve an old version of the file **fil.pro**, which is located in ***$IDLLIB/old*** and was entered into the IDL Library system (and so into CVS) on 1999-12-15.

```
cd $IDLLIB
cvs checkout -D "1999-12-15" -p $IDLLIB/old/fil.pro > ./fil.pro
```

These steps extract the desired version into the file **fil.pro** in directory ***$IDLLIB***. This is done so as not to overwrite the current version in its home directory. Then, from ***$IDLLIB***, the file may be sent to the user by whatever means seems appropriate. After the file has been sent, it must be deleted from ***$IDLLIB***.

7.0 SUGGESTIONS FOR FURTHER WORK

There are a number of things that may be done to enhance the IDL Library, making it more useful for the user and simplifying the tasks of the IDL Librarian. Many of these will only come to light after the system begins being used, and in fact the HTML documents read by people using the IDL Library encourage them to submit suggestions.

However, in writing and setting up the IDL Library system and writing this report a number of potential enhancements have suggested themselves. Due to time constraints, however, they could not be implemented.

Suggested enhancements include:

- A routine should be written that would be run on all files to be included in the library to ensure that there are no non-text characters in the files.
- Scripts (either shell or Perl) could be written to simplify the CVS duties of the IDL Librarian. These scripts could:
  - reduce IDL Librarian procedures to a one-step command;
  - perform the safety and sanity checks on the input that the procedures in Sections 5.2 and 6.0 do not perform; and
  - perform automatic handling of the ";>>IDLLIB Date:" information tag data.
- Code could be written to make greater use of the information tag data, for example searches by keywords, owner, etc. could be performed.
- **MakeFI.pl** could be modified so that it works on individual directories or files rather than the entire contents of the IDL Library. This would speed up the IDL Librarian's tasks once the size of the Library becomes fairly large.
- A script could be written to check files for the non-existence of comment blocks; it could either insert dummy blocks or flag groups of comment lines for possible conversion to comment blocks when comment blocks are missing.
- A script could be written to check new files for the presence of information tags and insert them if they are missing.
- The search screens could have a checklist of items to be displayed in the search results.

P516704.PDF [Page: 22 of 31]

APPENDIX A. INFORMATION TAGS

IDL code and referral files may all contain information tags, flags that tell **MakeFI.pl** that the line has information that may be extracted for Library maintenance purposes. In fact referral files are required to have at least one of these tags, as is indicated in Section 4.2 of this document.

The following table lists the tags, either sample values or the required optional values, and an indication as to whether the tags are file oriented, routine oriented, or both. Tags that are file oriented are generally more important for our purposes than the routine oriented tags except that the ";>>Name:" tag will be used to identify comment blocks that will be presented to users when a routine search is done.

No.	Tag	Sample Values or Options	File Use	Routine Use
1	;>>IDLLIB Date:	Date of the file's placement into the IDL Library in the CSA standard format, e.g. "2000-10-26"; there may only be one of these tags in a file	✓	
2	;>>Access:	• for downloadable code: "Download" • for referral routines; the e-mail address of the person who is to be contacted	✓	
3	;>>Files Needed:	Space delimited list of extra files that this file or routine needs to run, e.g. files containing called routines or "help" files. If this goes over one line, each line must have the tag.	✓	
4	;>>Owner:	Name of the person or group that owns the file, e.g. "Jim Theriault"	✓	✓
5	;>>Last Update:	Date of the file or routine's last update in the CSA standard format, e.g. "2000-10-26"	✓	✓
6	;>>Version:	Version number in whatever format the owner is using, if any	✓	✓
7	;>>Keywords:	Space delimited list of keywords of use in describing the file or routine or its capabilities. If this goes over one line, each line must have the tag.	✓	✓
8	;>>Name:	Name of the routine		✓

As might be expected, there are a number of rules associated with how these tags may be used:

P516704.PDF [Page: 23 of 31]

- The tags will only be processed if they appear with a comment block. See Section 3.0 of this report for a description of a comment block.
- Tags may not be preceded on a line by anything but whitespace, and it is preferred that they start in column 1.
- Tags may appear in any order.
- In all cases, the tag must be entered as written in the above table, except that the text is not case sensitive.
- Tags that appear in the same comment block are assumed to be associated, and each comment block is treated as a separate entity.
- Tags may be present with no value, which is the same as if the tag was not present.
- Tag values may only contain the characters **0-9**, **a-z**, **A-Z**, **space**, **hyphen**, and **underscore** except for a referral's e-mail address, which will also contain **@** and **periods**, and version numbers, which may also contain **periods**.
- The ";>>IDLLIB Date:" tag must contain the date that the file is put into the IDL Library system.
- If a comment block has tags but lacks the ";>>Name:" tag (or has one with no value), then the comment block is taken to be the file's comment block, and not a routine's.
- The ";>>Access:" tags may only be given one of the values specified.
- If there is no IDL code in a file (i.e. it consists only of comments) it must have an ";>>Access:" value of an e-mail address.
- If a file has IDL code the ";>>Access:" tag will be set to "`Download`" in **file.info** regardless of what it may be set to in the file.
- Tags ";>>Files Needed:" and ";>>Keywords:" may appear more than once in a comment block if the number of entries associated with them goes over a line. If any of the other tags appear more than once in a comment block, only the value of the last appearance will be noted.
- If a file-only comment block lacks ";>>Owner:" data and all the routines in the file have the same ";>>Owner:" value, then that value will be associated with the file as well.
- If file-only tags appear in a routine's comment block (as indicated by the presence of the tag ";>>Name:"), the file-only tag's data will be ignored if the items have already been declared for the file. If the tags have not been declared in a file-only block then they will be used for the file.
- Anything that looks like a tag but isn't in the table above (e.g. ";>>Size:") will be ignored.

As **MakeFI.pl** is processing **.pro files**, it will declare a file to be in error under the following circumstances.

- If it does not have one and only one ";>>IDLLIB Date:" tag.
- If the tag ";>>Name:" appears in a file with code, and there is no code naming a PRO or FUNCTION with the tag's value.
- If more than one comment block appears with the same ";>>Name:" tag value.
- If more than one PRO or FUNCTION section of code has the same routine name.

P516704 PDF [Page: 24 of 31]

APPENDIX B. MakeFI.pl AND file.info CONTENTS

The Perl script **MakeFI.pl** must be run by the IDL Librarian to create the file **file.info**, which contains information on all the **.pro** files from directories below ***$IDLLIB***. The program runs through the files noting frequently required information and putting this information into **file.info**, which is essentially the "index card file" of the library.

file.info consists of two types of lines of data, "file lines" and "routine lines." File lines always starts things off, begin with the string "FILE", and contain information about the file in a series of fields. If the file contains any routines, then the file line will be followed by one routine line for each of the PRO or FUNCTION routines in the file. Each routine line will start with the "P", "F", or "H" and will contain data fields with information on that routine.

The fields in each line are separated by a double colon (i.e. "::"), and if a field should have more than one value, the individual values are separated by plus signs (i.e. "+").

Any time there are no data for a field or the information is unknown, a zero fills the field.

In the fields' descriptions below, any time that the data are obtained from information tags, the tag is specified in brackets at the end of the description.

The fields in the file lines are:

"FILE"	- a text string containing exactly that string
filename	- the name of the file
directory	- the directory the file is in relative to ***$IDLLIB***
IDL Library date	- the date that the file was put into the IDL Library system; this is used as the date index if a user wishes to retrieve an old version of a file from CVS
file type	- "C" if the file is a code file and "H" if it is a referral file ("H" because the file would usually be the header of a code file)
comment block line	- a pair of "+"-separated numbers with the first number indicating the line number of the first line of a comment block in the file (i.e. the line after ";+") and the second number indicating the last line of the comment block (i.e. the line before ";-"); if there is more than one comment block for the file the pairs of values for each comment block are separated by "+"
access	- either "D" if the file is downloadable code, or the e-mail address of the person the file can be obtained from if the file is a referral [;>>Access:]
owner	- the name of the individual or group who owns the file [;>>Owner:]
last update	- the date of the last file update [;>>Last Update:]
version	- the file's version number [;>>Version:]
keywords	- a series of keywords relating to the file [;>>Keywords:]

P516704.PDF [Page: 25 of 31]

files needed	- the names of any files needed to run the program (e.g. those containing called routines or help files) [;>>Files Needed:]

The fields in the routine lines are:

type	- in a code file either "P" if the routine is of type PRO or "F" if it is a FUNCTION; "H" for a pure comment block file ("H" is because the comment block is usually the routine's header and to prevent "C" being confused with code)
name	- the routine's name [;>>Name:]
comment block line	- a pair of "+"-separated numbers with the first number indicating the line number of the first line of the routine's comment block in the file (i.e. the line after ";+") and the second number indicating the last line of the comment block (i.e. the line before ";-")
owner	- the name of the individual or group who owns the routine [;>>Owner:]
last update	- the date of the last routine update [;>>Last Update:]
version	- the routine's version number [;>>Version:]
keywords	- a series of keywords relating to the routine [;>>Keywords:]

P516704.PDF [Page: 26 of 31]

APPENDIX C. USER ROUTINES

This appendix presents a brief description of the routines that a user can encounter while using the IDL Library system. Users always go from HTML to HTML document, however in so doing they often cause Perl CGI scripts to produce these documents on the fly.

The following chart lists the documents, scripts, and links that a user may encounter, with routines that are run invisibly by the user enclosed in parentheses. Indented routines are run by those above it with a lesser indentation, and items at a common indentation may all be displayed simultaneously. The HTML documents in the list are located in directory ***$IDLLIB/html*** and the Perl scripts are located in directory ***$IDLLIB/CGI***. The chart is followed by a description of the items in the chart.

1. **IDL_Library_Home.html**
2. **filesearch.html**
3. **(filsrch.pl)**
4. **(frhtml.pl)**
5. download link
6. e-mail link
7. **routsearch.html**
8. **(routsrch.pl)**
4. **(frhtml.pl)**
5. download link
6. e-mail link
9. **textsearch.html**
10. **(txtsrch.pl)**
4. **(frhtml.pl)**
5. download link
6. e-mail link
11. IDL Librarian e-mail link
12. RSI link
13. Fanning Consulting link

1. **IDL_Library_Home.html** - presents the Library's home page, describing what can be done, indicating how to contact the IDL Librarian, and describing the types of submissions that can be made to the Library.
2. **filesearch.html** - is called when the user wishes to search for a specific file or list all of them. It accepts either a complete or partial filename and can perform either case sensitive or case insensitive filename searches. Alternatively, entering "*" for the filename causes all the **.pro** files to be listed. The user's data are entered into a form, and clicking this page's "Search" button starts the Perl script **filsrch.pl**.
3. **(filsrch.pl)** - searches through **file.info** for the specified filename and writes the results to the screen as an HTML document. If files were

P516704.PDF [Page 27 of 31]

found then a table will list the filenames along with their owners, access, and notes. If the file has any comment blocks the filenames will be links to **frhtml.pl**; if the files are code then the access will be the linked words "Download" or that and "Download All" (see "5. download link"); if the file is a referral file then the note will be the linked word "Referral" (see "6. e-mail link").

4. **(frhtml.pl)** - called as a link to files in tables of search results created by **filsrch.pl**, **routsrch.pl**, or **txtsrch.pl** when files have comment blocks. This routine writes to an HTML document (and so to the screen) the comment blocks of the specified file. If there are routines in the file with their own comment blocks, the individual routines' names are listed in a table of contents at the top of the HTML listing and are randomly accessible.
5. download link - created as a link to tar/gzip versions of the file found by **filsrch.pl**, **routsrch.pl**, or **txtsrch.pl**. The link to "Download" is to the listed file itself, whereas the "Download All" link is to a file containing the listed file plus all other files named in the listed file's ";>>Files Needed:" information tags.
6. e-mail link - created as a link to the user's default e-mail program if the file found by **filsrch.pl**, **routsrch.pl**, or **txtsrch.pl** is a referred file. When clicked it opens a new mail message to the e-mail address associated with the ";>>Access:" information tag, inserting the name of the file as the e-mail's subject line.
7. **routsearch.html** - is called when the user wishes to search for a specific routine (based on either a complete or partial routine name) or list all of them (by entering "*" as a search string). The user's data are entered into a form, and clicking this page's "Search" button starts the Perl script **routsrch.pl**.
8. **(routsrch.pl)** - searches through **file.info** for the specified routine and writes the results to the screen as an HTML document. If matching routines were found then a table will list the routines along with their owners, the file that the routine is in, the file's access, and notes. If the file has any comment blocks the filenames will be links to **frhtml.pl**; if the files are code then the access will be the linked words "Download" or that and "Download All" (see "5. download link"); if the file is a referral file then the note will be the linked word "Referral" (see "6. e-mail link").
9. **textsearch.html** - is called when the user wishes to perform either case sensitive or case insensitive searches for text strings in the **.pro** files. The user's data are entered into a form, and clicking this page's "Search" button starts the Perl script **txtsrch.pl**.
10. **(txtsrch.pl)** - searches through all the **.pro** files listed in **file.info** for the specified strings and writes the results to the screen as an HTML document. If files were found to contain the strings, then a table will list the files along with the files' owners, access, and notes. If the file has any comment blocks, the filenames will be links to **frhtml.pl**; if the files are code then the access will be the linked words "Download" or that and "Download All" (see "5.

P516704.PDF [Page: 28 of 31]

download link"); if the file is a referral file then the note will be the linked word "Referral" (see "6. e-mail link").

11. IDL Librarian e-mail link - is a link to the user's default e-mail program that initiates an e-mail message to the IDL Librarian.
12. RSI link - is a link to the home page of Research Systems Inc., the producer of IDL.
13. Fanning Consulting link - is a link to a company that has a wealth of IDL related information, including FAQs and sample code.

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1 ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e g Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8)

xwave Solutions
5475 Spring Garden Road, Suite 601
Halifax, NS B3J 3T2

2 SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable)

UNCLASSIFIED

3 TITLE (the complete document title as indicated on the title page Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title)

DREA IDL Librarian User's Guide (U)

4 AUTHORS (Last name, first name, middle initial If military, show rank, e g Doe, Maj John E)

C. Calnan

5 DATE OF PUBLICATION (month and year of publication of document)	6a NO OF PAGES (total containing information Include Annexes, Appendices, etc)	6b NO OF REFS (total cited in document)
September 2001	24 (Approx.)	0

7 DESCRIPTIVE NOTES (the category of the document, e g technical report, technical note or memorandum If appropriate, enter the type of report, e g interim, progress, summary, annual or final Give the inclusive dates when a specific reporting period is covered)

CONTRACTOR REPORT

8 SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development Include address)

Defence Research Establishment Atlantic
PO Box 1012
Dartmouth, NS, Canada B2Y 3Z7

9a PROJECT OR GRANT NO (if appropriate, the applicable research and development project or grant number under which the document was written Please specify whether project or grant)	9b CONTRACT NO (if appropriate, the applicable number under which the document was written)
1cm	W7707-0-8211

10a ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity This number must be unique to this document)	10b OTHER DOCUMENT NOs (Any other numbers which may be assigned this document either by the originator or by the sponsor)
DREA CR 2000-167	

11 DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12 DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected)

Unlimited

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

13 ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual)

Over the years a not inconsiderable amount of IDL code has been written to perform data analysis and to produce graphic output. However, the vast majority of the IDL code has been written almost in isolation, with the various programmers generally working with little contact with one another. The result is that the same programs or functions are continually being rewritten as programmers are unaware that programs that they working on have already been written.

To circumvent this duplication of effort a contract was awarded to produce a library of IDL code. After consultation with various IDL programmers and considering the options, it was decided to make the library available via the DREA Intranet, and searchable in a number of ways. This was done.

Creating a library, however, is only the start of the process. The ongoing work involved with any library is maintenance, wherein the contents are kept up to date and new entries are made accessible to users as soon as possible. This manual describes the tasks required of an IDL Librarian in order to keep the library current.

14 KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title)

Software Library
Graphics
Signal Processing
IDL
Interactive Display Library

P516704 PDF [Page: 31 of 31]

Defence R&D Canada
is the national authority for providing
Science and Technology (S&T) leadership
in the advancement and maintenance
of Canada's defence capabilities.

R et D pour la défense Canada
est responsable, au niveau national, pour
les sciences et la technologie (S et T)
au service de l'avancement et du maintien des
capacités de défense du Canada.

#516704
CA020033

DEFENCE R&D DÉFENSE

www.drdc-rddc.dnd.ca